MQL – User's Guide

MQL is developed for substructure matching. Recently it contains plugins for CDK, but plugins for every other toolkit can be easily included.

These code-snippets give you an idea how to integrate MQL into your CDK Project:

```
01: MQLMatcher controller=null;
02:   try {
03:         controller = MQLMatcher.getInstance();
04:   } catch (PluginInitializeException e) {
05:         e.printStackTrace();
06:   }
07: String queryString = "c$1:c:c:c:c:c$:1";
08: MQLQuery query = new MQLQuery();
09: try {
10:         MQLParser.parseQuery(queryString, query);
11:   } catch (ParseException e) {
12:         e.printStackTrace();
13:   } catch (TokenMgrError e) {
14:         e.printStackTrace();
15:   }
16: query.initialize();
17: String smilesstring ="C1(CC2=CC=CC=C2)=CC=CC=C1";
18: SmilesParser smilesparser = new SmilesParser();
19: CDKMoleculeWrapper molecule = null;
20: HydrogenAdder hadder;
21: try {
22:         Molecule mol = smilesparser.parseSmiles(smilesstring);
23:         IsotopeFactory ifac = IsotopeFactory.getInstance();
24:         ifac.configureAtoms(mol);
25:         hadder = new HydrogenAdder(new ValencyChecker());
26:         hadder.addHydrogensToSatisfyValency(mol);
27:
28:         molecule = new CDKMoleculeWrapper(mol);
29:   } catch (Exception e) {
30:         e.printStackTrace();
31:   }
32: controller.initializeQuery(query);
33: controller.initializeMolecule(molecule);
34: UllmanMatchingAlgorithm matchingAlgo = new UllmanMatchingAlgorithm();
35: matchingAlgo.getUniqueMatchings(query,molecule,controller);
```

Lines 01-06 create an instance of an MQLMatcher. MQLMatcher is a loads the plugins, which are responsible for the matching. Lines 07-16 initialize the query. The query is a graph, the static parser processes the querystring. Lines 17-31 creates a molecule wrapper for CDK Toolkit molecules. Line 32 initializes the query within the matcher, line 33 the moleculeWrapper. Line 34 and 35 perform the matching. Please consult the javadoc for different types of matching routines.